SAMPLAN — A new tool to support effective SG sampling plans

Aaron M Bevill, Robert Binner, and Claude F Norman

International Atomic Energy Agency, Vienna, Austria

a.bevill@iaea.org

ABSTRACT

Nuclear safeguards experts use advanced statistical algorithms to calculate verification sampling plans. Sampling plans must meet or exceed safeguards performance targets with minimal use of safeguards resources. For example, a plan will call for a certain number of resource-sparing non-destructive assay (NDA) verifications and high-precision destructive analysis (DA) samples in order to achieve an overall detection probability with minimal "cost" of inspection time, shipping fees, laboratory burden, facility disruption, etc.

In recent years, experts in safeguards statistical and probabilistic methodologies have proposed algorithmic advances to further improve sampling plans' efficiency while improving field usability, adapting to revised safeguards objectives, and maintaining a high level of effectiveness. These algorithms are implemented in the SAMPLAN software toolkit. Today, SAMPLAN is notable for both its current capabilities and its development strategy: evolutionary prototyping is used to rapidly refine algorithms, user interfaces, test cases, etc., which can then be implemented in production software for long-term use. This paper documents the prototyping strategy and SAMPLAN's resulting capabilities.

Keywords: IAEA; safeguards; diversion detection; detection probability; methods development

1 INTRODUCTION

The "timely detection of diversion of significant quantities of nuclear material … and deterrence of such diversion by the risk of early detection" is a key safeguards objective for the International Atomic Energy Agency (IAEA) [1]. In practice the IAEA meets this objective through nuclear material accountancy, backed by inspections to verify the accountancy declarations' accuracy. Since it is impossible to verify every declared item with perfect sensitivity, the inspections achieve this safeguards objective—deterring diversion by creating a risk of early detection—using random sampling. Drawing robust safeguards conclusions from random samples requires bespoke statistical algorithms.

For example, bespoke algorithms are used to evaluate the detection probability (DP) or conversely to plan sample sizes sufficient to achieve a specified DP. The most commonly used sample size calculation (SSC) algorithm is an approximation introduced in an era in which "not all the inspectors had ready access to a [personal computer] on all occasions" [2]. Although "the quality of the approximate algorithm was found to be very good" [ibid.], new approaches have been developed to improve upon this SSC algorithm and its implementation. The resulting SSC process is more flexible and precise and therefore yields plans that are equally effective but more resource-efficient. The updated SSC algorithms, software prototype, and process are discussed in the *SAMPLAN CAPABILITIES* section below.

Besides **what** these capabilities are, readers may also take interest in **how** these capabilities were developed from whiteboard sketches to field-ready prototypes. The SAMPLAN prototyping framework was carefully selected to overcome the challenges one observes in projects integrating new technology into existing safeguards systems. These challenges and the SAMPLAN prototyping framework are discussed next.

2 REFINING STATISTICAL ALGORITHMS FOR SAFEGUARDS

For safeguards algorithms and software, one is confronted by technology integration challenges as early as the requirements gathering phase. The new technology is required in some ways to maintain continuity with the replaced technology while in other ways improving upon it. For example, facility approaches may specify total sample sizes but leave flexibility in the verification methods applied to each sample. It is often difficult to assess *a priori* which aspects can be improved and which must be maintained; the assessment usually depends on interdependent processes, which have evolved over decades and may vary by facility type and scale and by safeguards agreement. In such projects, even carefully elicited requirements may be incomplete or misadjusted.

SAMPLAN is one such project with limited *a priori* knowledge of requirements. The predecessor software and processes for the typical SSC problem are frequently used, and training on these processes is well established. However, the software and algorithms have not been updated since at least 2001 [2], and many improvements have been proposed. Given the complexity of IAEA safeguards, experts have limited ability to foresee side-effects of updated SSC algorithms, e.g. sample sizes exceeding facility-specific targets or insufficient data to effectively assess declaration-measurement quality. SSC calculations are also typically performed in the field under time pressure, so the system must minimize computation time and user interface complexity. Distilling these considerations into precise requirements is not possible without field testing.

To field test new SSC algorithms and methods, SAMPLAN was created as an evolutionary prototype—flexible enough for rapid improvement but reliable enough for frequent use. Algorithms and procedures are proposed by methodological experts, then refined in a design–build–test–review feedback loop. Experience from the refining process is captured into requirements, test cases, and documentation. Success of the prototyping exercise depends on how well this experience is captured, how much the prototype improves in each iteration, and how rapidly iteration can occur.

To strike the optimal balance between flexibility and reliability, the evolutionary prototype spans the gap between research artifacts (demonstration scripts, white papers, spreadsheets) and enterprise software. See comparison in Table I.

The evolutionary prototype currently supports both routine evaluations and advanced use cases. Refer to the system dependency diagram in Fig. 1. Users (safeguards inspectors and some evaluators) are assumed to not have scripting experience, so routine evaluations are supported by a graphical user interface (GUI). The interface is implemented using a R *Shiny* library [3]. *Shiny* usually creates web applications, but has been adapted here to self-update and launch on the user's personal computer. Screenshots from the GUI are included in later sections.

TABLE I. Comparison between research artifacts, evolutionary prototypes, and enterprise software.

	Research Artifacts	**Evolutionary Prototype**	**Enterprise Software**
Code quality	Often messy; evolves as features are needed	Code begins messy, stabilizes, then cleaned	Clean
Testing & Data QC	Limited, as needed	Testbed to identify key QC and tests	Comprehensive coverage
Documentation	Limited, as needed	Detailed documentation added to cover process and algorithms	Comprehensive coverage
Version control & peer review	Limited, as needed	Increasingly formal as field testing expands	Formal
Framework	Researcher choice (Excel, VBA, Python, SAS, Matlab, R, …)	Scripting language module with extension to compiled languages, SQL, GUI as needed	Enterprise standards (e.g. formal C#, .net, SQL Server projects)
Maintenance Life	Temporary proof of concept	Dozens of iterations (until requirements stabilize)	Stable long-term use

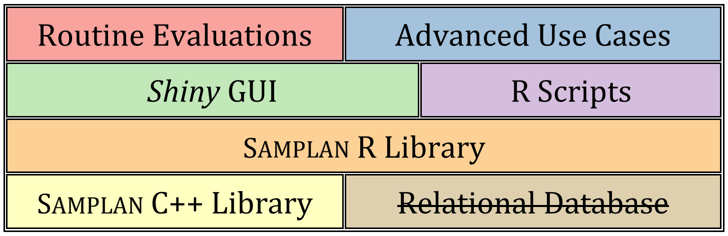


Figure 1. A conceptual diagram depicting how evaluations are supported by high-level libraries, which in turn are supported by low-level libraries. Relational databases are not used in SAMPLAN at this time.

In addition to the GUI, evaluators can address advanced use cases using R, e.g. scripts, R Markdown files, and the command line. The scripting interface is particularly powerful for repeated function calls (as one often needs in sensitivity studies and algorithm testing). The language R is primarily supported (to facilitate collaboration with statistical experts), but evaluators have also demonstrated Python use via *rpy2* [4].

Both the GUI and the scripts are built on the algorithms implemented in the SAMPLAN R library. The library is deployed via an internally hosted package repository. **SAMPLAN prototypers can easily implement, locally test, and deploy small improvements in under an hour.**

As functionality in the SAMPLAN R library matures, some portions are rewritten in C++. C++ is more difficult to write and maintain than R, but vastly outperforms R for operations that cannot be vectorized. Therefore not all functionality is planned to be migrated to C++; good candidates for the C++ library are functions that are computationally intensive, difficult to vectorize within R, and feature-stable.

When features are migrated to C++, the analogous R implementation is usually maintained as well. The C++ implementation is preferred for routine use; the redundant R implementation facilitates expert review and can be rapidly adapted for further research. The two implementations are kept in sync using test cases, and maintenance effort is minimal for stable functionality.

One notes that pairing R with C++—or more broadly, pairing a flexible scripting language with an optimized compiled language—is a design pattern common across disciplines. For example, engineering modeling/simulation codes use Python to drive compiled C++ and Fortran modules [5], often with script access to the Python library [6][7]. Similarly, many statistics and machine learning toolkits are developed as Python or R wrappers for C++ libraries [8][9][10][11][12]. To support projects like these, various toolkits have emerged to facilitate C++ interoperability with Python [13][14][15] and R [16][17]. SAMPLAN is hardly the first project to use this design pattern, and it is able to leverage a large collection of open-source tools to streamline prototype development.

Alongside the SAMPLAN C++ library, a relational database could be used to stabilize and scale certain operations. A lightweight internal database instance would be useful to prototype the database schema needed for an enterprise implementation. Alternatively, the R code could run read-only queries on externally maintained databases to collect parameters for SAMPLAN calculations (as is done in other evolutionary prototypes maintained by the authors). Neither database approach has been necessary for SAMPLAN prototyping yet. However, the SAMPLAN R library uses database-like table structures to facilitate later conversion to a formal database.

As Table I suggests, SAMPLAN is now maturing even as new methods are added. Test- and documentation-coverage are rapidly rising, and the version control system was recently upgraded from automated snapshots to a *git* [18] repository. Functionality is migrated to (or removed from) the GUI and C++ library in response to user feedback.

Through testing on dozens of real-world use cases, powerful new safeguards capabilities have been created.

3 SAMPLAN CAPABILITIES

SAMPLAN functions like a typical desktop application: The user can launch the GUI with a click; updates are automatically installed (if the package repository can be reached); inputs can be saved and loaded; integrated demonstration files can be loaded to review training cases.

Within this framework, DP and SSC modules applicable to various safeguards use cases can be implemented. Three modules are currently implemented:

- an overhaul of the *Basic SSC*, also known as the *Nested Sampling Plan*, used for most safeguards sample size calculations;
- the new *Multi-Stratum DP* algorithm [19], to evaluate DP for diversion scenarios spanning multiple strata, implemented in support of the State-Level Approach Improvement Project (SLAIP) [20]; and
- the new *TRIPS* algorithm [21], to plan and evaluate randomly scheduled inspections, also implemented in support of the SLAIP.

An additional two modules are planned:

- an update of the *Two-Stage SSC*, for verification methods that randomly select a number of items, then randomly select a number of sub-items from each selected item and
- an update of the *Follow-up SSC*, to determine verification actions in the event that a defective item is identified.

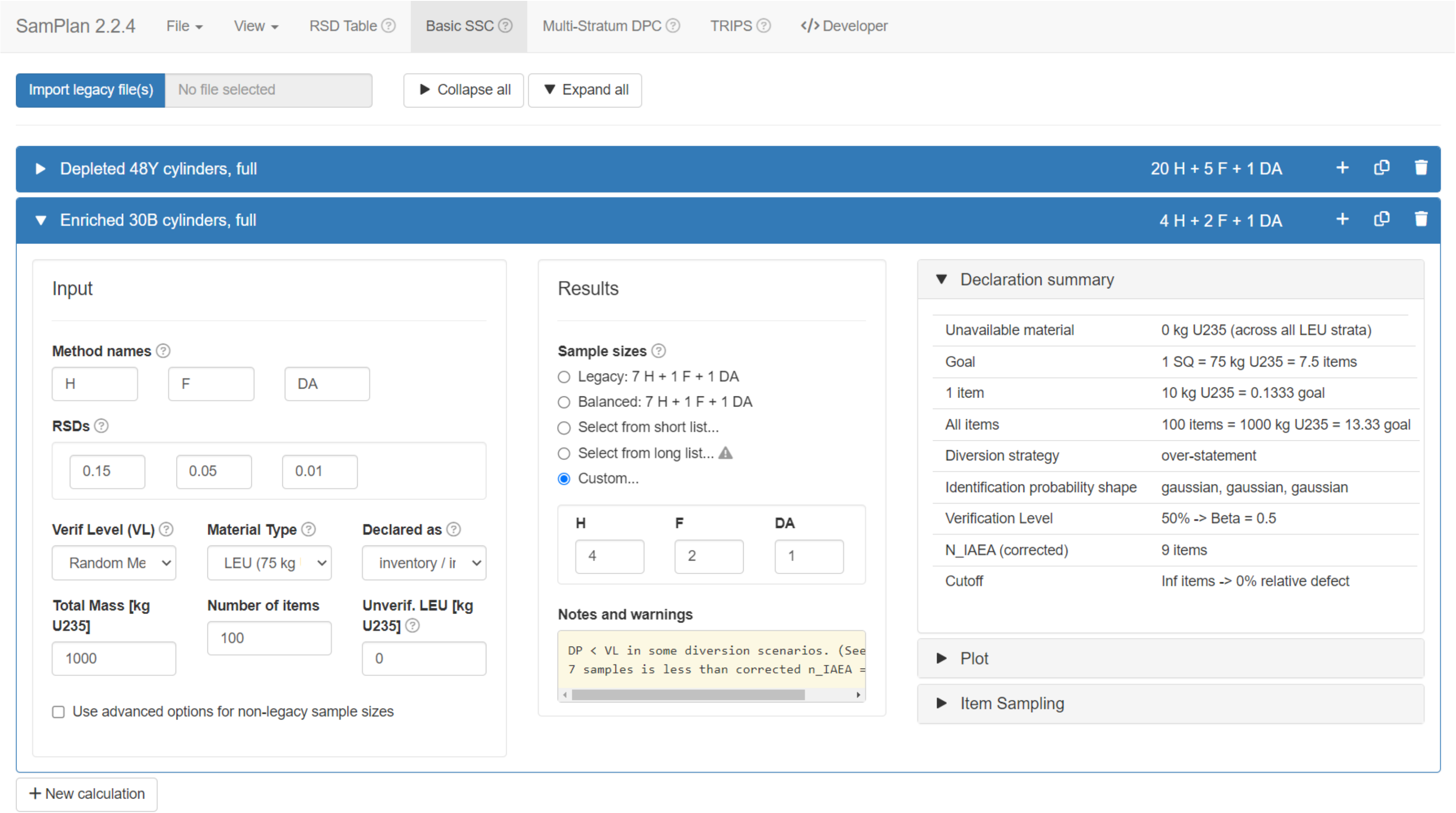
The predecessor software implemented these capabilities as separate applications, but their integration in SAMPLAN allows the modules to leverage capabilities from one another. For example, the Multi-Stratum DP module now uses the strata and settings specified in the Basic SSC module. This means that the advanced features of the Basic SSC module (described below) automatically flow into multi-stratum calculations.

Of the three currently implemented modules, only the Basic SSC has not been described in earlier publications, so its significantly expanded capabilities are discussed in depth here.

3.1 "Basic" SSC Capabilities in SAMPLAN

The Basic SSC is most commonly used for safeguards. It models a scenario in which up to three methods of varying precision are used to verify a stratum of multiple similar objects; refer to Refs. [19] for further details. The GUI for this SSC is shown in Fig. 2.

On the *Input* panel, users enter information about the stratum being verified. Refer to the list of fields in Table II. These fields are analogous to fields in the predecessor SSC software, and SAMPLAN can import predecessor files. However, the SAMPLAN fields have been adapted to avoid manual calculations. For example, inspectors currently must calculate *beta* by looking up the verification level (usually *RH*, *RM*, or *RL*), interpreting it as a probability (0.9, 0.5, or 0.2), then subtracting from one (0.1, 0.5, or 0.8). SAMPLAN simply requires the verification level. The SAMPLAN fields and the predecessor equivalent are compared in Table II. **Crucially, SAMPLAN minimizes the manual calculations necessary during time-sensitive inspections.**



▶ Depleted 48Y cylinders, full — 20 H + 5 F + 1 DA

▼ Enriched 30B cylinders, full — 4 H + 2 F + 1 DA

Input

Method names: H, F, DA

RSDs: 0.15, 0.05, 0.01

Verif Level (VL)	Material Type	Declared as
Random Me	LEU (75 kg	inventory / ir

Total Mass [kg U235]	Number of items	Unverif. LEU [kg U235]
1000	100	0

☐ Use advanced options for non-legacy sample sizes

Results

Sample sizes

○ Legacy: 7 H + 1 F + 1 DA
○ Balanced: 7 H + 1 F + 1 DA
○ Select from short list...
○ Select from long list... ⚠
● Custom...

H	F	DA
4	2	1

Notes and warnings

```
DP < VL in some diversion scenarios. (See
7 samples is less than corrected n_IAEA =
```

▼ Declaration summary

Unavailable material	0 kg U235 (across all LEU strata)
Goal	1 SQ = 75 kg U235 = 7.5 items
1 item	10 kg U235 = 0.1333 goal
All items	100 items = 1000 kg U235 = 13.33 goal
Diversion strategy	over-statement
Identification probability shape	gaussian, gaussian, gaussian
Verification Level	50% -> Beta = 0.5
N_IAEA (corrected)	9 items
Cutoff	Inf items -> 0% relative defect

▶ Plot

▶ Item Sampling

Figure 2. An overview of the S*AMPLAN Basic SSC GUI. For the* Enriched 30B cylinders *stratum, the* Input, Results, *and* Declaration Summary *panels are shown. The* Plot *panel is collapsed here but will be shown in a later figure. Image has been edited to remove confidentiality warnings and reduce whitespace.*

TABLE II. A comparison of Basic SSC inputs (SAMPLAN vs predecessor).

SAMPLAN Field	**Description**	**Predecessor Equivalent**
RSDs	Relative Standard Deviation: Estimated uncertainty of each verification instrument. RSD is expressed as a decimal (not percent) in keeping with safeguards conventions.	Same as SAMPLAN, but labeled *delta*.
Verification Level	Detection Probability required by the state-level approach, usually *Random High (90%)*, *Random Medium (50%)*, and *Random Low (20%)*. Arbitrary percentages can be entered for other cases.	Inspector must calculate *beta*: one minus the verification level (e.g. *0.1*, *0.5*, or *0.8*).
Material Type	One of *LEU*, *HEU*, *Pu*, etc, which defines the Significant Quantity (SQ) amount and measurand (U mass, ^{235}U mass, Pu mass, etc). Arbitrary amounts can be entered for unforeseen cases.	Inspector must enter the divertor's *goal* corresponding to 1 SQ: *75* (kg ^{235}U) for low-enriched uranium, *20000* (kg U) for depleted uranium, etc.
Declared as	Whether the material is declared as inventory, outbound shipments, or inbound receipts.	Inspector must select *Und* for receipts and *Ovr* for inventory and shipments.
Total Mass	The total mass of the specified isotope can usually be copied from the Itemized Inventory Listing.	Inspector must calculate and enter the mass **per item**.
Number of items	The number of declared items in the stratum.	Same as SAMPLAN, but labeled *N*.
Unverified mass	The mass in the stratum that is inaccessible for verification. Per safeguards procedure [23], the user sums all unverifiable mass with the same *Material Type*. QC ensures that unverified mass is consistent across strata.	No equivalent field; the inspector must manually total the unverifiable material across strata and subtract it from the "goal" (see above).

In addition to these predecessor-like input fields, new fields have been added. For example, the new "Balanced" algorithm can use non-Gaussian measurement error models, e.g. a step function of the number of defects. Most users do not need to use these advanced options; they can simply leave the *advanced options* box unchecked and use widely applicable defaults.

From these inputs, SAMPLAN calculates sample sizes for each instrument. SAMPLAN offers multiple sampling plans because there is no single "best" plan. Any plan that achieves the required DP is considered valid, and the inspector's preference among these depends on various operational considerations. Therefore the inspector may choose to use the plan calculated by the “legacy” algorithm (which replicates the predecessor software), one of multiple new constrained optimization algorithms (to be described in upcoming publications), or even enter a custom plan. Entering a custom plan allows the user to check its effectiveness in terms of DP, which is important for planning around instrument failures and for post-inspection DP evaluation. **This DP-calculation capability was not possible with the predecessor inspector software**; a separate DP application was used by evaluators.

High-priority notes and warnings are listed below the sampling plan, and additional diagnostics are available at the right edge of the GUI. The *Declaration Summary* echoes input information about the number of items, SQ definition, etc., which helps the user confirm that the inputs were correctly figured and entered.

Below the *Declaration Summary*, the DP is plotted across a spectrum of diversion scenarios. See Fig. 3. This plot is very important because it shows whether the selected plan is effective. (The

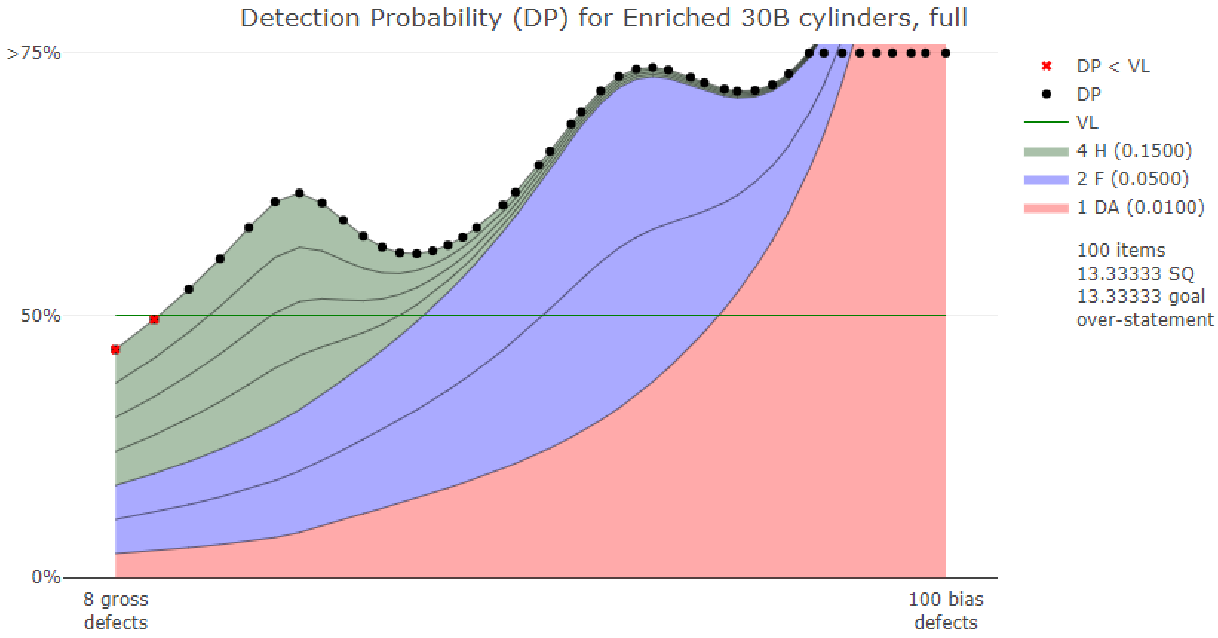


Figure 3. A detail of the Plot portion of the Samplan Basic SSC GUI. The black markers plot the achieved DP (vertical axis, transformed so $\log(1-DP)$ is linear) for each considered diversion scenario. The diversion scenario is characterized by the number of defective items from which material is diverted (horizontal axis, logarithmic scale). A red X indicates that the achieved DP is less than the required verification level (green line), and therefore the plotted sampling plan is not effective. The shaded contours estimate the DP if each H (green), F (blue), and DA (red) sample were sequentially removed. In this example, the contours indicate that all three methods are equally effective for gross defects (left edge), but methods H and eventually F lose effectiveness as the number of defects increases. Image has been edited to remove confidentiality warnings.

plan is valid if the red/black DP points are all above the green VL line.) Similar plots are used in the predecessor software and SSC training, but in SAMPLAN the plot is interactive and exportable. Importantly, the SAMPLAN DP plot introduces a new "nonlinear y-axis" transformation that makes the DP appear to "stack up" as samples are added. The thickness of each sample indicates its effectiveness for that diversion scenario. **With this visual guide, the user can rapidly find a custom plan that is both effective, efficient and, practical.**

In addition to these improvements, various experimental features have been demonstrated in the SAMPLAN Basic SSC for research purposes. For example, a correction has been proposed to deduct the false-alarm rate from the DP, and a stochastic DP calculation has been demonstrated to account for covariance effects. These capabilities are not intended for safeguards use (yet), but their implementation in SAMPLAN will enable testing on real use cases.

4 CONCLUSIONS

SAMPLAN is an evolutionary prototype for DP and SSC calculations, spanning the gap between early plans and enterprise software development. The algorithms are implemented in R and C++ but can be used via a GUI or scripting interface. The prototype uses off-the-shelf open-source frameworks and well-established design patterns. This allows revisions to be implemented and fielded within hours, not days, so the new methods can rapidly mature.

To date, three modules have been prototyped in SAMPLAN: a new multi-stratum DP algorithm, planning/evaluation algorithms for randomly scheduled inspections, and improvements of the routinely used SSC. This paper focuses on the latter, and highlights three improvements. First, inputs have been reformulated to minimize the need for manual calculations in the field. Second, inspectors can now assess alternative sampling plans in the field. These alternative sampling plans can be calculated with advanced settings, including non-Gaussian error models. Third, improved DP visualization enables inspectors to rapidly improve alternative plans. These improvements will enable inspectors to use safeguards resources as efficiently as possible while continuing to conduct effective verification activities.

5 ACKNOWLEDGEMENTS

The authors wish to acknowledge IAEA safeguards evaluator Sandrine Cormon for demonstrating Python access of the SAMPLAN R library.

6 REFERENCES

[1] International Atomic Energy Agency (1972). The Structure and Content of Agreements Between the Agency and States Required in Connection with the Treaty on the Non-proliferation of Nuclear Weapons (INFCIRC 153, corrected). Vienna, Austria.

[2] IAEA, "Statistical Concepts and Techniques for IAEA Safeguards," IAEA SG-PR-2016 Rev. 5, 1998.

[3] Chang W, et al. (2023). shiny: Web Application Framework for R. R package version 1.7.4.9002, <https://shiny.rstudio.com/> accessed 2023-03-22.

[4] Rpy2 - R in Python. <https://rpy2.github.io/> accessed 2023-03-21.

[5] VERA 4.1 (RSICC code package CCC-855). <https://rsicc.ornl.gov/codes/ccc/ccc8/ccc-855.html> accessed 2023-03-22.

[6] ADVANTG 3.2.0 (RSICC code package CCC-831). <https://rsicc.ornl.gov/codes/ccc/ccc8/ccc-831.html> accessed 2023-03-22.

[7] PyNE: The Nuclear Engineering Toolkit. <https://github.com/pyne/pyne> accessed 2023-03-22.

[8] PyTorch. <https://pytorch.org/> accessed 2023-03-22.

[9] TensorFlow. <https://www.tensorflow.org/> accessed 2023-03-22.

[10] Keras: Deep Learning for humans. <https://keras.io/> accessed 2023-03-22.

[11] Riddell, A., Hartikainen, A., & Carter, M. (2021). PyStan (3.0.0). <https://pypi.org/project/pystan> accessed 2023-03-22.

[12] RStan. <https://github.com/stan-dev/rstan> accessed 2023-03-22.

[13] Boost.Python. <https://www.boost.org/doc/libs/1_81_0/libs/python/doc/html/index.html> accessed 2023-03-22.

[14] SWIG. <https://www.swig.org/> accessed 2023-03-22.

[15] Cython: C-Extensions for Python. <https://cython.org/> accessed 2023-03-22.

[16] RTools: Toolchains for building R and R packages from source on Windows. <https://cran.r-project.org/bin/windows/Rtools/> accessed 2023-03-22.

[17] Eddelbuettel, D., et al. Rcpp: Seamless R and C++ Integration version 1.0.10. <https://cran.r-project.org/web/packages/Rcpp/index.html> accessed 2023-03-22.

[18] Git. <https://git-scm.com/> accessed 2023-03-22.

[19] Bevill, A., et al (2021). Multi-stratum Detection Probability Calculations for IAEA Safeguards: Foundations and Early Progress. Proceedings of the INMM & ESARDA Joint Annual Meeting.

[20] Sylvester, K. (2021). Experience Gained in the State Level Approach Improvement Project. Presented at INMM & ESARDA Joint Annual Meeting.

[21] Krieger, T., et al (2021). Random Inspection Planning for Misuse Detection in Safeguards. Proceedings of the INMM & ESARDA Joint Annual Meeting.

[22] Krieger, T. et al (2017). "Statistical Methods for Verification Sampling Plans," IAEA STR-381 draft, 2017.

[23] Safeguards Criteria: Procedures for Sampling Plans. IAEA document SG-SC-Annex-06 version 2, 2009-09-02. See specifically paragraph 3.2.